LLNL-CONF-457551

Using the Schur Complement to Reduce Runtime in KULL's Magnetic Diffusion Package

T. Brunner, T. Kolev

September 29, 2010

Nuclear Explosives Code Developer's Conference 2010
Los Alamos, CA, United States
October 18, 2010 through October 22, 2010

Lawrence Livermore National Laboratory

Using the Schur Complement to Reduce Runtime in KULL's Magnetic Diffusion Package (U)

Nuclear Explosives Code Developers' Conference 2010

October 18-22, 2010

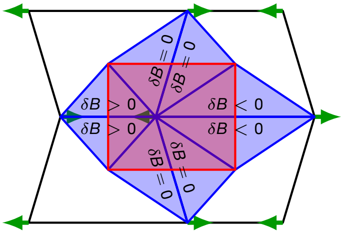
Thomas A. Brunner Tzanio V. Kolev

Special thanks to Tom Gardiner (SNL)

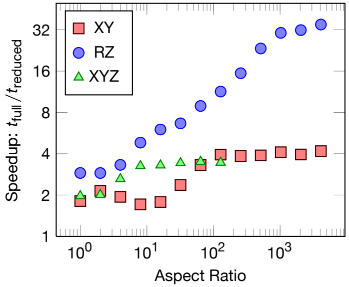
Lawrence Livermore National Laboratory, P. O. Box 808, Livermore, CA 94551
This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

LLNL-CONF-457551

A new MHD package has been added to KULL



Kull's hydro uses subzonal stresses



- The new MHD package is based on the standard edge finite-element discretization
- To match compatible hydro, our fields are subzonal
  - Electric fields on edges of sub-tet mesh
  - Method works on arbitrary polyhedral meshes
  - This gives us many unknowns
- Reducing the matrix with a Schur complement always improves runtimes
  - Speedup improves as mesh gets more distorted

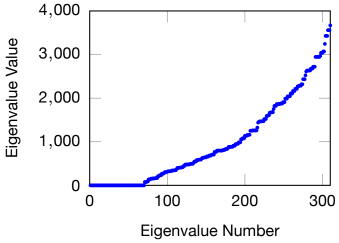
Magnetic diffusion heats material and dissipates field

Discretize **E** with edge finite elements:

$$\mathbf{E} = \sum_e E_e \mathbf{w}^e$$

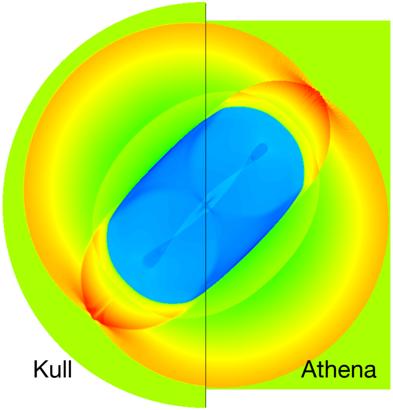
$$\nabla \times \frac{\Delta t}{\mu_0} \nabla \times \mathbf{E}^{n+1} + \overline{\overline{\sigma}} \cdot \mathbf{E}^{n+1} = \nabla \times \frac{\mathbf{B}^n}{\mu_0}$$

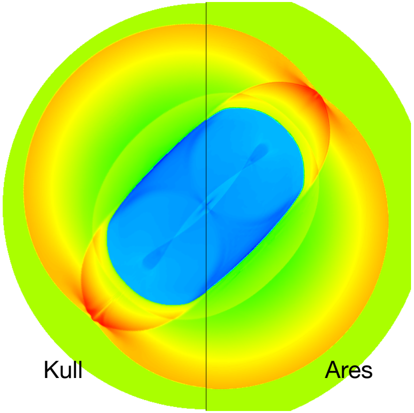
$$\mathbf{B}^{n+1} = \mathbf{B}^n - \Delta t \nabla \times \mathbf{E}^{n+1}$$



- This equation for the current is (nearly) singular
- In pure void, $\sigma = 0$ and $\mathbf{J} = \sigma \mathbf{E} = 0$, so **E** unconstrained
- Use CG with *hypre*'s Auxiliary-space Maxwell Solver as preconditioner
  - Solves two, easier, nodal problems and projects to edge problem
  - In void, we add the constraint $\mathbf{E} = \nabla \phi$ to make it non-singular
- Using pure void is more robust than using a small σ
  - A small conductivity is resolution dependent: $\sigma_{\text{small}} \sim 1/\Delta x$
  - But extra work for void is about 10% slower than using a floor

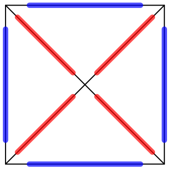
KULL performs well on ideal MHD tests



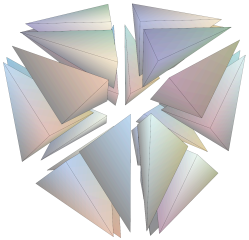


- Wave structure and instability agree with others
- We work fully unstructured, arbitrary polyhedral meshes
- Mesh becomes highly distorted, with angles nearing 180°

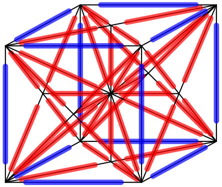
Robustness and generality is paid for with unknowns



6 vs. 2 edges/zone (RZ)



Each hex has 24 tets

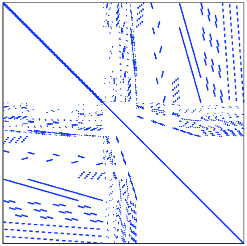


29 vs. 3 edges/zone (3D)

- Relative to a standard quad or hex discretization (in blue)
  - In 2D-XY, we have twice the unknowns
  - In 2D-RZ, we have thrice the unknowns
  - In 3D, we have 10 times the unknowns

Can we trade extra *local* work for some of the expensive *global* HYPRE solve to improve runtime?

During matrix assembly, we reduce global unknowns


Total 3D matrix **A**

- Full matrix is sum of tet matrices

$$\mathbf{A} = \sum_t \mathbf{A}_t$$

- We form groups of tets into clumps

$$\mathbf{A} = \sum_c \mathbf{A}_c \quad \text{with} \quad \mathbf{A}_c = \sum_{t_c} \mathbf{A}_{t_c}$$

- Edges are interior to the clump or on the boundary

$$\mathbf{A}_c \mathbf{x}_c = \mathbf{y}_c \quad \rightarrow \quad \begin{bmatrix} \mathbf{A}_{ii} & \mathbf{A}_{ib} \\ \mathbf{A}_{bi} & \mathbf{A}_{bb} \end{bmatrix} \begin{bmatrix} \mathbf{x}_i \\ \mathbf{x}_b \end{bmatrix} = \begin{bmatrix} \mathbf{y}_i \\ \mathbf{y}_b \end{bmatrix}$$

- Isolated interior edges are eliminated with Schur complement

$$\left(\mathbf{A}_{bb} - \mathbf{A}_{bi} \mathbf{A}_{ii}^{-1} \mathbf{A}_{ib} \right) \mathbf{x}_b = \mathbf{y}_b - \mathbf{A}_{bi} \mathbf{A}_{ii}^{-1} \mathbf{y}_i \quad \rightarrow \quad \mathbf{A}_r \mathbf{x}_b = \mathbf{y}_r$$

Reduced unknowns are recovered locally after global solve

- Now, we sum reduced, local clump matrices into global matrix

$$\mathbf{A}_R = \sum_c \mathbf{A}_{r_c}, \quad \mathbf{y}_R = \sum_c \mathbf{y}_{r_c}$$

- Then we can solve global system with *hypre*

$$\mathbf{x}_B = \mathbf{A}_R^{-1}\mathbf{y}_R$$

  - Reduced matrix is much smaller than full matrix
  - $\mathbf{A}_R$ is a discretization on the clump mesh
- For each clump, exactly solve for the interior unknowns

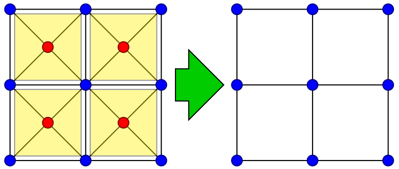
$$\mathbf{x}_i = \mathbf{A}_{ii}^{-1}\mathbf{y}_i - \mathbf{A}_{ii}^{-1}\mathbf{A}_{ib}\mathbf{x}_b$$

- We trade more cache-efficient, dense linear algebra ($\mathbf{A}_{ii}^{-1}$) for less global work computing $\mathbf{A}_R^{-1}$

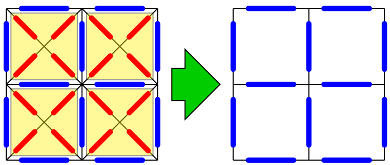
What triangles or tets do we choose to make the clumps?

In 2D we eliminate the zone-interior unknowns

Clumps are defined by the triangles that make up an original mesh zone



Reduce XY unknowns to mesh nodes

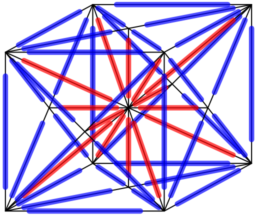


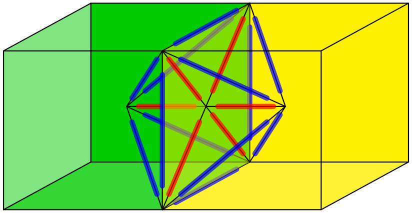
Reduce RZ unknowns to mesh edges

Mesh Type	rows/hex	matrix entries/hex	matrix entries/row
XY original	2	14	7
XY quad	1	9	9
RZ original	6	30	5
RZ quad	2	14	7

- Number of unknowns and matrix entries are lower.
  - But matrix is less sparse
- We have recovered the same number of unknowns and nonzeros as the standard quad discretizations
  - But the discretization is not the same

In 3D we must think outside the box


The tetrakis hexahedron is the obvious clump of tets


The octahedron that spans each face is a much better clump of tets

- Reduction increases matrix bandwidth
- Good performance tied more to matrix size than unknowns

Mesh Type	rows/hex	matrix entries/hex	matrix entries/row
“tet’d” hex (original)	29	461	16
tetrakis hexahedron	15	1107	74
octahedron	11	335	30
standard hex	3	99	33

A tetrakis hexahedron is a non-regular icositetrahedron (24-sided polyhedron) formed by adding square pyramids to the faces of a hexahedron. Eric W. Weisstein, *Mathworld*, `http://mathworld.wolfram.com/TetrakisHexahedron.html`

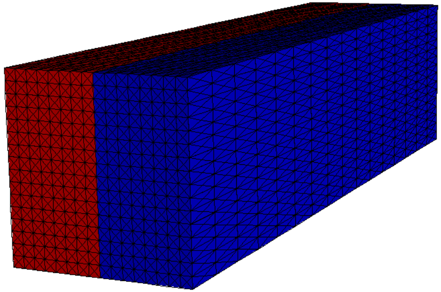
The reduced matrix has nice mathematical properties

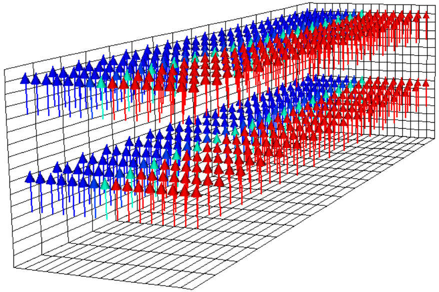
We have detailed proofs of these statements, which we'll skip...

- Many of the properties of the original matrix carry over to the reduced matrix
  - The reduced matrix is sparse, unlike most Schur complements
  - We have a large (near) null space, so we need AMS
  - The AMS preconditioner works on the reduced matrix
  - In 2D the reduced matrix has the same graph as a standard quad discretization
- Some of the properties are much better for the reduced matrix
  - The condition number is lower
  - The ratio between the strongest and weakest off-diagonals in the matrix is better, making it easier for AMS/AMG to make good choices about eliminating entries
  - In 2D the reduced matrix is even nicer than the standard quad discretization
- It is as if we discretized directly on the reduced mesh
  - But we get solutions for *all* of the original unknowns.

See Brunner and Kolev, "Alegraic multigrid for linear systems obtained by explicit element reduction," submitted to *SIAM Journal on Scientific Computing*, July 2010

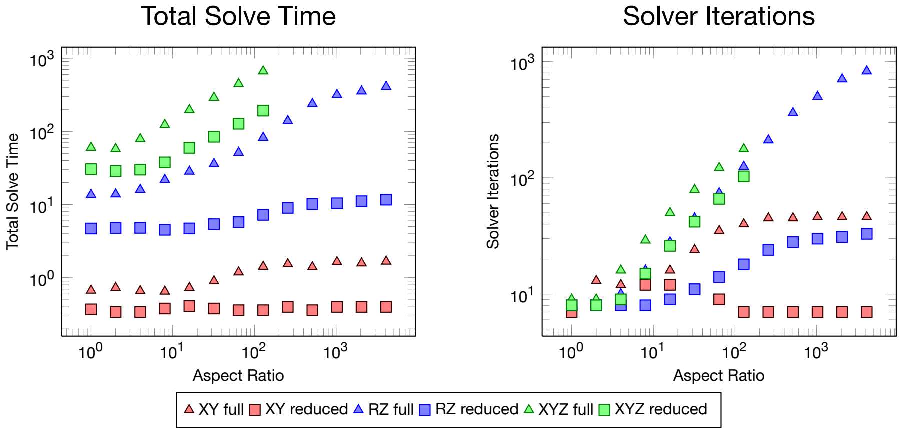
Test 1: Conductivity jump and varying aspect ratio


The mesh and conductivities


B-field solution

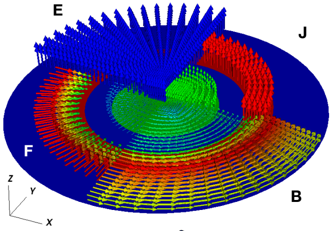
- A magnetic field diffuses from a void region into material
- The mesh is stretched to create zones with high aspect ratios, keeping resolution in the interesting direction fixed
- We compare run times and iteration counts for 2D-XY, 2D-RZ, and 3D-XYZ geometries

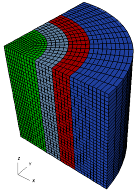
Test 1: Speedup improves as aspect ratio increases

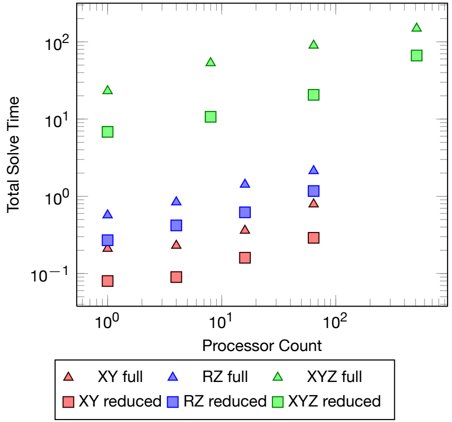


- Solve times are always faster with the reduced matrix
  - XY: 1.7-4.2$\times$, XYZ: 2.0-3.5$\times$, RZ: 2.9-35$\times$!
- Improvement from smaller matrix and reduced iteration count
- Setup is faster, despite extra work

Solve time improves on larger, realistic problems



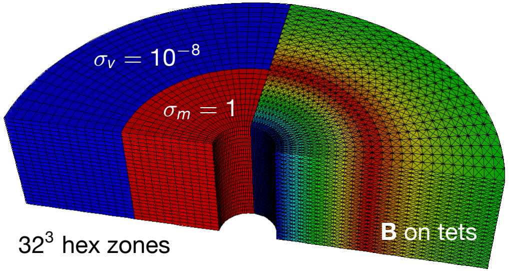




- Increase problem size with processors to 10M unknowns
- XY simulation is the top plane; RZ is the front plane
- Solve time improvement is independent of problem size.
- Reduction reduces solve time between 1.8-4.4×

See Gardiner and Brunner, “Interesting & Useful MHD Test Problems,” JOWOG-42, April 2010, SAND2010-2509-P

Comparison of hex-only to sub-tet'd discretizations



- Run the same problem multiple ways
  - Kull: tet'd hex
  - Kull: reduced matrix
  - Ares: pure hex
- Both Kull methods solve same system
- *hypre* used to solve both

Mesh	rows	matrix entries	code setup	*hypre* solve	total time	iters
Tet (K)	969k	15.2M	18.5	107.0	125.5	13
Reduced (K)	367k	10.9M	12.9	33.5	46.4	9
Hex (A)	105k	3.3M	5.9	16.4	22.3	18
Ratio (K/A)	*9.3*	*4.6*	*2.2*	*2.0*	*2.1*	*0.5*

- Kull runtime 2× slower for 9.3× more unknowns
  - Need to run convergence study, plotting error vs. runtime
- Condition number improves from hex to tet to reduced

Conclusions

- We've developed a new subzonal MHD discretization that works with Kull's hydro
  - Supports arbitrary polygonal and polyhedral meshes
  - Works on with pure void more robustly than small conductivity
  - But it has more unknowns than zonal discretization
- We combine unknowns using the Schur complement
  - Reduced matrix is still sparse, and works with AMS
  - In 2D, we recover the number of unknowns as a quad discretization
  - Reduced matrix has better properties, making it easier to solve
  - Setup is faster, despite extra work, since we touch less memory
- Overall run time is improved between 2 and 30 times!
  - Improvement increases as aspect ratio gets worse